EAST Search History

Ref #	Hits	Search Query	DBs	Default Operator	Plurals	Time Stamp
L4	325	712/209.ccls.	US-PGPUB; USPAT	OR	OFF	2006/03/08 18:18
L3	15	(US-20030093775-$ or US-20030093774-$ or US-20030093649-$ or US-20040073892-$ or US-20010047438-$).did. or (US-6243668-$ or US-5577233-$ or US-6704925-$ or US-6163764-$ or US-4347565-$ or US-5029069-$ or US-5680568-$ or US-6973417-$ or US-5870575-$ or US-6457117-$). did.	US-PGPUB; USPAT	OR	OFF	2006/03/08 18:18
S141	1	(translat$4 with (predictable unpredictable) with operand).clm.	US-PGPUB; USPAT	OR	ON	2006/03/08 13:16
S140	1	(translat$4 with ((operand adj (using setting)))).clm.	US-PGPUB; USPAT	OR	ON	2006/03/08 13:16
S139	2	(resume adj translation (resumetranslation)).clm.	US-PGPUB; USPAT	OR	ON	2006/03/08 13:14
S138	4	(translat$4 with flag with set$4 with clear$4).clm.	US-PGPUB; USPAT	OR	ON	2006/03/08 13:13
S137	6	(translat$4 with suspend$4 with resum$4).clm.	US-PGPUB; USPAT	OR	ON	2006/03/08 13:11
S136	1	(translat$4 with operand with us$4 with value with determin$4).clm.	US-PGPUB; USPAT	OR	OFF	2006/03/08 13:10
S135	1	(translat$4 with operand with set$4 with value with determin$4).clm.	US-PGPUB; USPAT	OR	OFF	2006/03/08 13:09
S134	2	(translat$4 with instruction with precedent with operand with flag). clm.	US-PGPUB; USPAT	OR	OFF	2006/03/08 13:09
S133	4	((dynamic adj emulation) with legacy with translat$4).clm.	US-PGPUB; USPAT	OR	OFF	2006/03/08 13:07
S130	2	S128 and tag	US-PGPUB; USPAT	OR	OFF	2006/03/08 13:05
S132	7	S131 and tag	US-PGPUB; USPAT	OR	OFF	2006/03/08 12:55
S131	14	(US-20030093775-$ or US-20030093774-$ or US-20030093649-$ or US-20040073892-$).did. or (US-6243668-$ or US-5577233-$ or US-6704925-$ or US-6163764-$ or US-4347565-$ or US-5029069-$ or US-5680568-$ or US-6973417-$ or US-5870575-$ or US-6457117-$). did.	US-PGPUB; USPAT	OR	OFF	2006/03/08 12:55

EAST Search History

S128	7	(US-20030093776-$).did. or (US-6142682-$ or US-6516295-$ or US-5560013-$ or US-5819063-$ or US-6163764-$ or US-4794522-$). did.	US-PGPUB; USPAT	OR	OFF	2006/03/08 12:51
S127	69	S126 and (tag with operand)	US-PGPUB; USPAT	OR	OFF	2006/03/08 12:46
S125	4832	(code instruction) adj (conversion translation mimic)	US-PGPUB; USPAT	OR	OFF	2006/03/08 12:46
S126	6896	(code instruction) adj (conversion translat$4 mimic)	US-PGPUB; USPAT	OR	OFF	2006/03/08 12:46
S124	555	S123 and translat$4	US-PGPUB; USPAT	OR	ON	2006/03/08 12:39
S123	816	S121 and S122	US-PGPUB; USPAT	OR	ON	2006/03/08 12:38
S122	7629	operand with (calculat$4 determin$6 detect$4 complet$4)	US-PGPUB; USPAT	OR	ON	2006/03/08 12:38
S121	1009	operand with tag	US-PGPUB; USPAT	OR	ON	2006/03/08 12:37
S120	384	opcode with tag	US-PGPUB; USPAT	OR	ON	2006/03/08 12:37
S119	386	(S118 S114) and (tag$4 index$4 mark$3)	US-PGPUB; USPAT	OR	ON	2006/03/08 12:30
S118	331	S115 or S116 or S117	US-PGPUB; USPAT	OR	OFF	2006/03/08 12:24
S117	161	717/138.ccls.	US-PGPUB; USPAT	OR	OFF	2006/03/08 12:24
S116	124	717/135.ccls.	US-PGPUB; USPAT	OR	OFF	2006/03/08 12:24
S115	85	717/134.ccls.	US-PGPUB; USPAT	OR	OFF	2006/03/08 12:24
S114	331	703/26.ccls.	US-PGPUB; USPAT	OR	OFF	2006/03/08 12:24
S113	1	"4638423".pn.	US-PGPUB; USPAT	OR	OFF	2006/03/08 12:23
S112	303	S111 and (translat$4 (cross adj compil$6) port$4)	US-PGPUB; USPAT	OR	OFF	2006/03/07 15:47
S111	331	S106 or S108 or S110	US-PGPUB; USPAT	OR	OFF	2006/03/07 15:02
S106	85	717/134.ccls.	US-PGPUB; USPAT	OR	OFF	2006/03/07 15:02
S110	161	717/138.ccls.	US-PGPUB; USPAT	OR	OFF	2006/03/07 14:59
S108	124	717/135.ccls.	US-PGPUB; USPAT	OR	OFF	2006/03/07 14:59

EAST Search History

S103	5	(US-6243668-$ or US-5577233-$ or US-4347565-$ or US-5029069-$ or US-5680568-$).did.	USPAT	OR	OFF	2006/03/07 14:57
S104	294	(abort stop suspend) same translat$4 same (flag indicator)	USPAT	OR	OFF	2006/03/07 14:54
S101	86	translat$4 with operand with set$4	USPAT	OR	OFF	2006/03/07 10:45
S102	42	S101 and (simulat$4 emulat$4 model$4)	USPAT	OR	OFF	2006/03/07 10:45
S99	21	(US-20040194070-$ or US-20030093776-$).did. or (US-4638423-$ or US-5301302-$ or US-5546552-$ or US-5560013-$ or US-5577231-$ or US-5577233-$ or US-5751982-$ or US-5790825-$ or US-5933622-$ or US-6009261-$ or US-6075937-$ or US-6142682-$ or US-6243668-$ or US-6516295-$ or US-6704925-$ or US-6785801-$ or US-5819063-$ or US-6163764-$ or US-4794522-$).did.	US-PGPUB; USPAT	OR	OFF	2005/06/24 18:06
S97	8	S89 and operand	US-PGPUB; USPAT	OR	OFF	2005/06/24 14:53
S98	10	S89 and (instruction with (size length))	US-PGPUB; USPAT	OR	OFF	2005/06/24 14:53
S96	11	S89 and byte	US-PGPUB; USPAT	OR	OFF	2005/06/24 14:52
S95	4	S89 and arrangement	US-PGPUB; USPAT	OR	OFF	2005/06/24 14:48
S89	20	(US-20040194070-$).did. or (US-4638423-$ or US-5301302-$ or US-5546552-$ or US-5560013-$ or US-5577231-$ or US-5577233-$ or US-5751982-$ or US-5790825-$ or US-5933622-$ or US-6009261-$ or US-6075937-$ or US-6142682-$ or US-6243668-$ or US-6516295-$ or US-6704925-$ or US-6785801-$ or US-5819063-$ or US-6163764-$ or US-4794522-$).did.	US-PGPUB; USPAT	OR	OFF	2005/06/24 14:48
S77	18	(US-20040194070-$).did. or (US-4638423-$ or US-5301302-$ or US-5546552-$ or US-5560013-$ or US-5577231-$ or US-5577233-$ or US-5751982-$ or US-5790825-$ or US-5933622-$ or US-6009261-$ or US-6075937-$ or US-6142682-$ or US-6243668-$ or US-6516295-$ or US-6704925-$ or US-6785801-$ or US-5819063-$).did.	US-PGPUB; USPAT	OR	OFF	2005/06/24 14:29

EAST Search History

S87	48	("5918056" "6052685" "6240417" "4954942" "5278962" "5392420" "5623673" "5682310" "5970237" "6128732" "6212614" "6212614" "6397242" "6397379" "6446094" "5642491" "5815727" "5937185" "5953520" "5325512" "5590342" "5706407" "5909696" "6223284" "6223284" "4591967" "4611286" "4794522" "4812981" "4851828" "4888688" "4954968" "4975872" "5063499" "5056013" "5226154" "5278961" "5289581" "5289587" "5325469" "5357628" "5369749" "5369767" "5390314" "5438674" "5440710" "5452454" "5455926" "5485614" "5530673").pn.	USPAT	OR	OFF	2005/06/24 12:04
S88	50	("5539901" "5566121" "5566326" "5598553" "5604864" "5615328" "5630052" "5636227" "5644755" "5682481" "5694582" "5710934" "5720015" "5732201" "5749094" "5764659" "5765206" "5774694" "5787493" "5793714" "5796984" "5796566" "5802052" "5805473" "5815686" "5819015" "5822784" "5832299" "5842011" "5852720" "5857074" "5862083" "5867096" "5896393" "5910876" "5913052" "5940850" "5965860" "5973964" "5982371" "5983309" "6049866" "6052524" "6052383" "6055651" "6063131" "6070224" "6078520" "6106565" "6115813").pn.	USPAT	OR	OFF	2005/06/24 12:04
S85	146	717/138.ccls.	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/06/24 12:03
S83	60	suspend with translation	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/06/24 11:43
S82	2	translation same (operand adj setting)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/06/24 11:38

EAST Search History

S81	71	S69 and (store with instruction)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/06/24 11:16
S69	230	(instruction adj set adj simulat$4)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/06/24 11:14
S79	146	translat$4 with (indirect in-direct) with address$4	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/06/24 11:12
S80	1	binary with translat$4 with (indirect in-direct) with address$4	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/06/24 11:12
S78	9	S77 and stream	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/06/24 11:10
S76	15	resume adj translation	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/06/24 10:34
S75	0	resume adj tranlation	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/06/24 10:34
S74	229	S73 and (simulat$4 emulat$4 model$4)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/06/24 10:33

EAST Search History

S73	379	S70 and S72	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/06/24 10:31
S72	2648	((in-direct indirect) adj address$4)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/06/24 10:30
S71	18	S69 and S70	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/06/24 10:27
S70	8291	(instruction byte operand) with align$8	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/06/24 10:17
S68	1	"09/992137"	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/06/24 10:16
S67	36	("5560013").URPN.	USPAT	OR	OFF	2005/06/24 09:59
S66	23	S64 and S65	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2005/06/24 09:53
S65	33	S63 and (emulation (instruction adj set adj simulat$4))	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2005/06/24 09:53
S64	61	S62 and (store with instruction) and (execution with (suspen$6 resum$5 stop$4 start$4))	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2005/06/24 09:52

EAST Search History

S63	168	S62 and (store with instruction)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2005/06/24 09:51
S62	802	"S/390"	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2005/06/24 09:51
S2	798	"S/390"	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2005/06/24 09:51
S61	14	(legacy with instruction) and (emulat$4) and (execution with (suspen$6 resum$5 stop$4 start$4))	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/06/24 09:35
S60	17	(US-20040194070-$).did. or (US-4638423-$ or US-5301302-$ or US-5546552-$ or US-5560013-$ or US-5577233-$ or US-5751982-$ or US-5790825-$ or US-5933622-$ or US-6009261-$ or US-6075937-$ or US-6142682-$ or US-6516295-$ or US-6704925-$ or US-6785801-$ or US-6243668-$ or US-5577231-$). did.	US-PGPUB; USPAT	OR	OFF	2005/06/23 18:27
S59	2	(dynamic adj object adj code adj translation).ti.	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/06/23 13:52
S58	15	S57 and modifi$6	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/06/22 21:09

EAST Search History

S57	17	(US-20040194070-$).did. or (US-4638423-$ or US-5301302-$ or US-5546552-$ or US-5560013-$ or US-5577233-$ or US-5751982-$ or US-5790825-$ or US-5933622-$ or US-6009261-$ or US-6075937-$ or US-6142682-$ or US-6516295-$ or US-6704925-$ or US-6785801-$ or US-6243668-$ or US-5577231-$). did.	US-PGPUB; USPAT	OR	OFF	2005/06/22 21:08
S56	194	S55 and (TLB with (size index))	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/06/22 20:32
S55	1206	(instruction with translation) and (TLB)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/06/22 20:32
S54	1	(instruction with translation) and (block adj tracking adj table)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/06/22 20:32
S11	1	"09/992130"	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/06/22 20:31
S53	17126	fujitsu.as.	USPAT	OR	OFF	2005/06/22 16:24
S52	190	amdahl.as.	USPAT	OR	OFF	2005/06/22 16:24
S51	7	(instruction with translat$5) and hotspot	USPAT	OR	OFF	2005/06/22 16:18
S50	1	("6516295").URPN.	USPAT	OR	OFF	2005/06/22 16:11
S49	23	legacy with instruction with translation	US-PGPUB; USPAT; USOCR	OR	OFF	2005/06/22 16:01
S48	110	translation adj index$5	US-PGPUB; USPAT; USOCR	OR	OFF	2005/06/22 16:00
S47	61	S16 and (translation with (flag set indicator))	US-PGPUB; USPAT; USOCR	OR	OFF	2005/06/22 15:58

EAST Search History

S46	5	S16 and (translation with (done complet$4) with (flag set indicator))	US-PGPUB; USPAT; USOCR	OR	OFF	2005/06/22 15:50
S16	715	S7 or S9	US-PGPUB; USPAT; USOCR	OR	OFF	2005/06/22 15:48
S45	82	S44 and S41	US-PGPUB; USPAT; USOCR	OR	OFF	2005/06/22 15:44
S44	581	(dynamic with translation) and index$5	US-PGPUB; USPAT; USOCR	OR	OFF	2005/06/22 15:44
S43	25	S15 and S41	US-PGPUB; USPAT; USOCR	OR	OFF	2005/06/22 15:43
S15	206	instruction adj set adj simulat$4	US-PGPUB; USPAT; USOCR	OR	OFF	2005/06/22 15:40
S42	172	((instruction with translation) and (index$5 with (block table) with translation))	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/06/22 15:31
S41	1192	((instruction with translation) and (block with translation))	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/06/22 15:31
S37	2551	(instruction and (block with translation))	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/06/22 15:30
S40	119	S39 and index with table	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/06/22 15:24
S39	470	S38 and table	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/06/22 15:24

EAST Search History

S38	545	S37 and emulat$4	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/06/22 15:24
S7	317	(703/26).CCLS.	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/06/22 15:22
S28	214	(instruction with translat$5 with (index flag table)) and (emulat$4)	US-PGPUB; USPAT; USOCR	OR	OFF	2005/06/22 14:26
S27	366	(instruction with translat$5 with (index flag table)) and (emulat$4 simulat$4 model$4)	US-PGPUB; USPAT; USOCR	OR	OFF	2005/06/22 14:03
S26	861	(instruction with translat$5 with (index flag table set)) and (emulat$4 simulat$4 model$4)	US-PGPUB; USPAT; USOCR	OR	OFF	2005/06/22 14:03
S25	18148	(translat$5 with (index flag table set)) and (emulat$4 simulat$4 model$4)	US-PGPUB; USPAT; USOCR	OR	OFF	2005/06/22 13:33
S21	190	S16 and flag	US-PGPUB; USPAT; USOCR	OR	OFF	2005/06/22 13:31
S23	1	S16 and (block with transform)	US-PGPUB; USPAT; USOCR	OR	OFF	2005/06/22 13:27
S24	1	S23 and address	US-PGPUB; USPAT; USOCR	OR	OFF	2005/06/22 13:26
S19	63	S16 and (table with index)	US-PGPUB; USPAT; USOCR	OR	OFF	2005/06/22 13:26
S22	11	S16 and translation with flag	US-PGPUB; USPAT; USOCR	OR	OFF	2005/06/22 13:13
S20	15	S16 and ((table with index) same translat$5)	US-PGPUB; USPAT; USOCR	OR	OFF	2005/06/22 13:13
S18	245	S17 and (translat$5)	US-PGPUB; USPAT; USOCR	OR	OFF	2005/06/22 12:24
S17	433	S16 and (table or index)	US-PGPUB; USPAT; USOCR	OR	OFF	2005/06/22 12:24

EAST Search History

S14	9	("4574344" | "4635188" | "4638423" | "4761733" | "5333287" | "5406644" | "5430862" | "5481693" | "5546552").PN.	US-PGPUB; USPAT; USOCR	OR	OFF	2005/06/22 11:45
S13	18	S12 and (store with instruction)	USPAT	OR	OFF	2005/06/22 10:30
S12	33	("4638423").URPN.	USPAT	OR	OFF	2005/06/22 10:29
S8	82	S7 and (instruction with translat$)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/06/22 10:22
S10	58	S9 and (instruction with translat$)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/06/22 10:17
S9	484	703/27.ccls.	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/06/22 10:17
S6	2	("5313614" | "5404478").PN.	US-PGPUB; USPAT; USOCR	OR	OFF	2005/06/21 12:05
S5	19	"S/390" with emulat$4	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2005/06/21 12:05
S4	116	S2 and emulat$4	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2005/06/21 12:04
S3	4	"S/390" with legacy with instruction	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2005/06/21 12:04

EAST Search History

S1	165	legacy with instruction with (translation emulat$4 simulat$4 execut$4)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2005/06/21 12:03

Terms used **instruction translation operand flag** Found **17,836** of **65,323**

Sort results by relevance | Save results to a Binder | Search Tips | Open results in a new window
Display results expanded form
Try an Advanced Search
Try this search in The ACM Guide

Results 1 - 20 of 200 Result page: **1** 2 3 4 5 6 7 8 9 10 next
Best 200 shown Relevance scale

1 Migrating a CISC computer family onto RISC via object code translation
Kristy Andrews, Duane Sand
September 1992 **ACM SIGPLAN Notices , Proceedings of the fifth international conference on Architectural support for programming languages and operating systems ASPLOS-V**, Volume 27 Issue 9
Publisher: ACM Press
Full text available: pdf(1.13 MB) Additional Information: full citation, references, citings, index terms

2 Native code compilation of Erlang's bit syntax
Per Gustafsson, Konstantinos Sagonas
October 2002 **Proceedings of the 2002 ACM SIGPLAN workshop on Erlang**
Publisher: ACM Press
Full text available: pdf(196.81 KB) Additional Information: full citation, abstract, references, citings

Erlang's bit syntax caters for flexible pattern matching on bit streams (objects known as *binaries*). Binaries are nowadays heavily used in typical Erlang applications such as protocol programming, which in turn has created a need for efficient support of the basic operations on binaries.To this effect, we describe a scheme for efficient native code compilation of Erlang's bit syntax. The scheme relies on *partial translation* for avoiding code explosion, an ...

3 Implications of structured programming for machine architecture
Andrew S. Tanenbaum
March 1978 **Communications of the ACM**, Volume 21 Issue 3
Publisher: ACM Press
Full text available: pdf(1.08 MB) Additional Information: full citation, abstract, references, citings, index terms

Based on an empirical study of more than 10,000 lines of program text written in a GOTO-less language, a machine architecture specifically designed for structured programs is proposed. Since assignment, CALL, RETURN, and IF statements together account for 93 percent of all executable statements, special care is given to ensure that these statements can be implemented efficiently. A highly compact instruction encoding scheme is presented, which can reduce program size by a factor of 3. Unlik ...

Keywords: computer architecture, computer organization, instruction set design,

machine architecture, program characteristics

4 A new approach to assembly software retargeting for microcontrollers
Ing-Jer Huang, Dao-Zhen Chen
January 2000 **Proceedings of the 2000 conference on Asia South Pacific design automation**
Publisher: ACM Press
Full text available: pdf(112.76 KB) Additional Information: full citation, references

5 Design methodologies for ASIPs: A novel approach for flexible and consistent ADL-driven ASIP design
Gunnar Braun, Achim Nohl, Weihua Sheng, Jianjiang Ceng, Manuel Hohenauer, Hanno Scharwächter, Rainer Leupers, Heinrich Meyr
June 2004 **Proceedings of the 41st annual conference on Design automation**
Publisher: ACM Press
Full text available: pdf(204.60 KB) Additional Information: full citation, abstract, references, index terms

Architecture description languages (ADL) have been established to aid the design of application-specific instruction-set processors (ASIP). Their main contribution is the automatic generation of a software toolkit, including C compiler, assembler, linker, and instruction-set simulator. Hence, the challenge in the design of such ADLs is to unambiguously capture the architectural information required for the toolkit generation in a single model. This is particularly difficult for C compiler and si ...

Keywords: ADL, ASIP, embedded processors

6 On the design of the local variable cache in a hardware translation-based java virtual machine
Hitoshi Oi
June 2005 **ACM SIGPLAN Notices , Proceedings of the 2005 ACM SIGPLAN/SIGBED conference on Languages, compilers, and tools for embedded systems LCTES'05**, Volume 40 Issue 7
Publisher: ACM Press
Full text available: pdf(118.36 KB) Additional Information: full citation, abstract, references, index terms

Hardware bytecode translation is a technique to improve the performance of the Java Virtual Machine (JVM), especially on the portable devices for which dynamic compilation is infeasible. However, since the translation is done on a single bytecode basis, it is likely to generate frequent memory accesses for local variables which can be a performance bottleneck.In this paper, we propose to add a small register file to the datapath of the hardware-translation based JVM and use it as a local variabl ...

Keywords: hardware-translation, java virtual machine, memory hierarchy

7 An instruction set and microarchitecture for instruction level distributed processing
Ho-Seop Kim, James E. Smith
May 2002 **ACM SIGARCH Computer Architecture News , Proceedings of the 29th annual international symposium on Computer architecture ISCA '02 , Proceedings of the 29th annual international symposium on Computer architecture ISCA '02**, Volume 30 Issue 2
Publisher: IEEE Computer Society, ACM Press

Full text available: pdf(1.08 MB) Publisher Site Additional Information: full citation, abstract, references, citings, index terms

An instruction set architecture (ISA) suitable for future microprocessor design constraints is proposed. The ISA has hierarchical register files with a small number of accumulators at the top. The instruction stream is divided into chains of dependent instructions (strands) where intra-strand dependences are passed through the accumulator. The general-purpose register file is used for communication between strands and for holding global values that have many consumers.A microarchitecture to supp ...

8 Binary translation

Richard L. Sites, Anton Chernoff, Matthew B. Kirk, Maurice P. Marks, Scott G. Robinson
February 1993 **Communications of the ACM**, Volume 36 Issue 2
Publisher: ACM Press
Full text available: pdf(4.84 MB) Additional Information: full citation, references, citings, index terms

Keywords: CISC computers, RISC computers, binary translation, computer architecture, processor architecture translation

9 Enhancing the performance of 16-bit code using augmenting instructions

Arvind Krishnaswamy, Rajiv Gupta
June 2003 **ACM SIGPLAN Notices , Proceedings of the 2003 ACM SIGPLAN conference on Language, compiler, and tool for embedded systems LCTES '03**, Volume 38 Issue 7
Publisher: ACM Press
Full text available: pdf(276.13 KB) Additional Information: full citation, abstract, references, citings, index terms

In the embedded domain, memory usage and energy consumption are critical constraints. Dual width instruction set embedded processors such as the ARM provide a 16-bit instruction set in addition to the 32-bit instruction set to address these concerns. Using 16-bit instructions one can achieve code size reduction and I-cache energy savings at the cost of performance. We have observed that throughout 16-bit Thumb code there exist Thumb instruction pairs that are equivalent to a single ARM instructi ...

Keywords: 16-bit thumb ISA, 32-bit ARM ISA, AX instructions, code size, embedded processor, instruction coalescing, performance

10 A practical method for code generation based on exhaustive search

David W. Krumme, David H. Ackley
June 1982 **ACM SIGPLAN Notices , Proceedings of the 1982 SIGPLAN symposium on Compiler construction SIGPLAN '82**, Volume 17 Issue 6
Publisher: ACM Press
Full text available: pdf(1.10 MB) Additional Information: full citation, abstract, references, citings, index terms

An original method for code generation has been developed in conjunction with the construction of a compiler for the C programming language on the DEC-10 computer. The method is comprehensive, determining evaluation order and doing register allocation and instruction selection simultaneously. It uses exhaustive search rather than heuristics, and is table-driven, with most machine-specific information isolated in the tables. Testing and evaluation have shown that the method is effective, tha ...

11 The UT1000 microprogramming simulator: an educational tool

F. Cornett
June 1989 **ACM SIGARCH Computer Architecture News**, Volume 17 Issue 4
Publisher: ACM Press
Full text available: pdf(574.05 KB) Additional Information: full citation, abstract, index terms

The concepts of microprogramming and firmware are frequently difficult to explain to computer science students. The subject of firmware is normally introduced in a beginning course of the computer science curriculum, usually as one of many terms for which a definition is memorized, but rarely understood. Even in advanced courses in computer organization or architecture, the concept may remain somewhat abstract to the student. Just as the practice of writing and executing programs in high level l ...

12 Binary translation and architecture convergence issues for IBM system/390

Michael Gschwind, Kemal Ebcioğlu, Erik Altman, Sumedh Sathaye
May 2000 **Proceedings of the 14th international conference on Supercomputing**
Publisher: ACM Press
Full text available: pdf(1.44 MB) Additional Information: full citation, abstract, references, index terms

We describe the design issues in an implementation of the ESA/390 architecture based on binary translation to a very long instruction word (VLIW) processor. During binary translation, complex ESA/390 instructions are decomposed into instruction "primitives" which are then scheduled onto a wide-issue machine. The aim is to achieve high instruction level parallelism due to the increased scheduling and optimization opportunities which can be exploited by binary translation software ...

13 A hardware architecture for implementing protection rings

Michael D. Schroeder, Jerome H. Saltzer
March 1972 **Communications of the ACM**, Volume 15 Issue 3
Publisher: ACM Press
Full text available: pdf(1.50 MB) Additional Information: full citation, abstract, references, citings

Protection of computations and information is an important aspect of a computer utility. In a system which uses segmentation as a memory addressing scheme, protection can be achieved in part by associating concentric rings of decreasing access privilege with a computation. This paper describes hardware processor mechanisms for implementing these rings of protection. The mechanisms allow cross-ring calls and subsequent returns to occur without trapping to the supervisor. Automatic hardware v ...

Keywords: Multics, access control, computer utility, hardware access control, protection, protection hardware, protection rings, segmentation, shared information, time-sharing, virtual memory

14 HPSm, a high performance restricted data flow architecture having minimal functionality

W. Hwu, Y. N. Patt
June 1986 **ACM SIGARCH Computer Architecture News , Proceedings of the 13th annual international symposium on Computer architecture ISCA '86**, Volume 14 Issue 2
Publisher: IEEE Computer Society Press, ACM Press
Full text available: pdf(875.61 KB) Additional Information: full citation, abstract, references, citings, index terms

Our recent work in microarchitecture has identified a new model of execution, restricted data flow, in which data flow techniques are used to coordinate out-of-order execution of sequential instruction streams. We believe that the restricted data flow model has great

potential for implementing very high performance computing engines. This paper defines a minimal functionality variant of our model, which we are calling HPSm. The instruction set, data path, timing and control of HPSm are all ...

15 Dynamic coalescing for 16-bit instructions
Arvind Krishnaswamy, Rajiv Gupta
February 2005 **ACM Transactions on Embedded Computing Systems (TECS)**, Volume 4 Issue 1
Publisher: ACM Press
Full text available: pdf(487.69 KB) Additional Information: full citation, abstract, references, index terms

In the embedded domain, memory usage and energy consumption are critical constraints.Embedded processors such as the ARM and MIPS provide a 16-bit instruction set, (called Thumb in the case of the ARM family of processors), in addition to the 32-bit instruction set to address these concerns. Using 16-bit instructions one can achieve code size reduction and instruction cache energy savings at the cost of performance. This paper presents a novel approach that enhances the performance of 16-bit Thu ...

Keywords: 16-bit Thumb ISA, 32-bit ARM ISA, AX instructions, Embedded processor, code size, energy, instruction coalescing, performance

16 HPSm, a high performance restricted data flow architecture having minimal functionality
Wen-Wei Hwu, Yale N. Patt
August 1998 **25 years of the international symposia on Computer architecture (selected papers)**
Publisher: ACM Press
Full text available: pdf(983.00 KB) Additional Information: full citation, index terms

17 A microprogrammed implementation of an architecture simulation language
William C. Hopkins, Gary Davidian
September 1977 **ACM SIGMICRO Newsletter , Proceedings of the 10th annual workshop on Microprogramming MICRO 10**, Volume 8 Issue 3
Publisher: IEEE Press, ACM Press
Full text available: pdf(916.63 KB) Additional Information: full citation, abstract, references, index terms

A "Machine Representation Language" (MRL), a tool for the evaluation and simulation of the instruction sets of computers, was designed for research in computer architecture. A novel hypothetical machine to perform the simulation uses an acyclic directed graph as its machine language. MRL requires an expandable associative memory and a recursive execution environment; the research requires extensive instrumentation of the simulation. A microprogrammed implementation satisfying th ...

18 The structure of yet another ALGOL compiler
H. Kanner, P. Kosinski, C. L. Robinson
July 1965 **Communications of the ACM**, Volume 8 Issue 7
Publisher: ACM Press
Full text available: pdf(1.54 MB) Additional Information: full citation, abstract, references, citings, index terms

A high-speed "top down" method of syntax analysis which completely eliminates "back-up" of the source string has been implemented in a convenient macro-language. A technique of simulation at compile time of the use of a conventional run-time stack enables the generation of code for expressions which minimizes stores, fetches and stack-

pointer motion at run time, while properly treating recursion and side effects of procedures. Block structure and recursion are handle ...

19 The architecture of the Burroughs B5000: 20 years later and still ahead of the times?
Alastair J. W. Mayer
June 1982 **ACM SIGARCH Computer Architecture News**, Volume 10 Issue 4
Publisher: ACM Press
Full text available: pdf(621.01 KB) Additional Information: full citation, abstract, references

The Burroughs B5000 was introduced over twenty years ago. The architectural features it introduced, and refined when it was upgraded to the B5500 and B6500, are only now appearing in new computer designs. This paper briefly describes some of these features, as they relate to high-level language and operating system support, and as interesting features in their own right. References are given for more detailed information.

20 Alpha AXP architecture
Richard L. Sites
February 1993 **Communications of the ACM**, Volume 36 Issue 2
Publisher: ACM Press
Full text available: pdf(4.62 MB) Additional Information: full citation, references, citings, index terms, review

Keywords: Alpha AXP chip

Results 1 - 20 of 200 Result page: 1 2 3 4 5 6 7 8 9 10 next

Terms used **operand calculation flag** Found **6,485** of **171,143**

Sort results by relevance Save results to a Binder Search Tips
Display results expanded form Open results in a new window

Try an Advanced Search
Try this search in The ACM Guide

Results 1 - 20 of 200 Result page: **1** 2 3 4 5 6 7 8 9 10 next
Best 200 shown Relevance scale

1 The user interface of GPSS/PC
Springer Cox
January 1984 **Proceedings of the 16th conference on Winter simulation**
Publisher: IEEE Press
Full text available: pdf(473.51 KB) Additional Information: full citation, abstract, citings, index terms

GPSS/PC is a new implementation of GPSS, the General Purpose Simulation System, with several interactive extensions to the user interface. Its overall design integrates a syntax directed statement parser with the run time system. This results in a number of advantages in the controllability of running simulations. Unlike traditional simulation language implementations, GPSS/PC was designed with the simulation primitives on the user interface. This means that functions previously ...

Keywords: GPSS

2 A hardware architecture for implementing protection rings
Michael D. Schroeder, Jerome H. Saltzer
March 1972 **Communications of the ACM**, Volume 15 Issue 3
Publisher: ACM Press
Full text available: pdf(1.50 MB) Additional Information: full citation, abstract, references, citings

Protection of computations and information is an important aspect of a computer utility. In a system which uses segmentation as a memory addressing scheme, protection can be achieved in part by associating concentric rings of decreasing access privilege with a computation. This paper describes hardware processor mechanisms for implementing these rings of protection. The mechanisms allow cross-ring calls and subsequent returns to occur without trapping to the supervisor. Automatic hardware v ...

Keywords: Multics, access control, computer utility, hardware access control, protection, protection hardware, protection rings, segmentation, shared information, time-sharing, virtual memory

3 A 32-bit CMOS microprocessor with six-stage pipeline structure
H. Kaneko, Y. Miki, S. Nohara, K. Koya, M. Araki
November 1986 **Proceedings of 1986 ACM Fall joint computer conference**
Publisher: IEEE Computer Society Press

Full text available: pdf(831.34 KB) Additional Information: full citation, references, citings, index terms

4 Denotational semantics of a calculator
Ken Slonneger
March 1993 **Proceedings of the 1993 ACM conference on Computer science**
Publisher: ACM Press
Full text available: pdf(489.14 KB) Additional Information: full citation, abstract, references, index terms

Denotational semantics1,2,3,4 provides a formal method of precisely and concisely specifying languages, programming languages as well as other kinds of software such as editors and file systems. But those attributes that make denotational descriptions so effective also lead programmers and software designers to reject the method as too complex. The main goal of this paper is to provide a nontrivial but relatively simple example suitable for presenting the fundamental co ...

5 Speculative dynamic vectorization
Alex Pajuelo, Antonio González, Mateo Valero
May 2002 **ACM SIGARCH Computer Architecture News , Proceedings of the 29th annual international symposium on Computer architecture ISCA '02 , Proceedings of the 29th annual international symposium on Computer architecture ISCA '02**, Volume 30 Issue 2
Publisher: IEEE Computer Society, ACM Press
Full text available: pdf(1.00 MB) Additional Information: full citation, abstract, references, index terms
Publisher Site

Traditional vector architectures have shown to be very effective for regular codes where the compiler can detect data-level parallelism. However, this SIMD parallelism is also present in irregular or pointer-rich codes, for which the compiler is quite limited to discover it. In this paper we propose a microarchitecture extension in order to exploit SIMD parallelism in a speculative way. The idea is to predict when certain operations are likely to be vectorizable, based on some previous history i ...

Keywords: Speculative dynamic vectorization, wide buses, speculative data computation, control independence, vector instructions

6 C Compiler Retargeting Based on Instruction Semantics Models
Jianjiang Ceng, Manuel Hohenauer, Rainer Leupers, Gerd Ascheid, Heinrich Meyr, Gunnar Braun
March 2005 **Proceedings of the conference on Design, Automation and Test in Europe - Volume 2**
Publisher: IEEE Computer Society
Full text available: pdf(186.41 KB) Additional Information: full citation, abstract

Efficient architecture exploration and design of application specific instruction-set processors (ASIPs) requires retargetable software development tools, in particular C compilers that can be quickly adapted to new architectures. A widespread approach is to model the target architecture in a dedicated architecture description language (ADL) and to generate the tools automatically from the ADL specification. For C compiler generation, however, most existing systems are limited either by the manu ...

7 VAX floating point: a solid foundation for numerical computation
Mary Payne, Dileep Bhandarkar
June 1980 **ACM SIGARCH Computer Architecture News**, Volume 8 Issue 4

Publisher: ACM Press
Full text available: pdf(979.78 KB) Additional Information: full citation, references

8 Migrating a CISC computer family onto RISC via object code translation
Kristy Andrews, Duane Sand
September 1992 **ACM SIGPLAN Notices , Proceedings of the fifth international conference on Architectural support for programming languages and operating systems ASPLOS-V,** Volume 27 Issue 9
Publisher: ACM Press
Full text available: pdf(1.13 MB) Additional Information: full citation, references, citings, index terms

9 Improving CISC instruction decoding performance using a fill unit
Mark Smotherman, Manoj Franklin
December 1995 **Proceedings of the 28th annual international symposium on Microarchitecture**
Publisher: IEEE Computer Society Press
Full text available: pdf(965.34 KB) Additional Information: full citation, references, citings, index terms

10 I-NET mechanism for issuing multiple instructions
L. Wang, C. L. Wu
November 1988 **Proceedings of the 1988 ACM/IEEE conference on Supercomputing**
Publisher: IEEE Computer Society Press
Full text available: pdf(978.12 KB) Additional Information: full citation, abstract, references, citings, index terms

Conventional instruction issuing methods use hardware control mechanism to issue instructions in multiple-functional-unit systems. They reach physical limitations due to the complexity of issuing logic when they intend to issue multiple instructions per cycle. A new method, I-NET, is presented in this paper to overcome this shortcoming. I-NET uses a post-compiler to detect the data dependencies among instructions. The detected data dependence is then attached to the instruction code to form ...

11 NICE: an elegant and powerful 32-bit architecture
B. Ulmann
September 1997 **ACM SIGARCH Computer Architecture News,** Volume 25 Issue 4
Publisher: ACM Press
Full text available: pdf(344.47 KB) Additional Information: full citation, abstract, index terms

The architecture described in the following articel is a direct successor of µ-EP-1 (cf. [1]) and was developed by the author and Robert Linden (Universität Bonn, FB Informatik). NICE is a 32-bit processor, utilizing a fixed instruction format, a register set of sixteen general purpose registers and sat extremely powerful but simple and easy to use instruction set which is supported by a variety of addressing modes. The smallest direct addressable data item in main memory is t ...

12 FLIP-FLOP: a stack-oriented multiprocessing system
Peter Grabienski
March 1991 **ACM SIGARCH Computer Architecture News,** Volume 19 Issue 1
Publisher: ACM Press
Full text available: pdf(672.87 KB) Additional Information: full citation, abstract, index terms

Many research and application fields are today computationally very demanding. This requirement has influenced the development of high-performance processors, vector processors and multiprocessor engines. This paper describes the development of the multiprocessing system FLIP-FLOP (Fast Link Periphery for a Forth Language Oriented Processor) which combines a stack oriented processor kernel and a communication coprocessor for message passing.In the past many muliprocessor systems based on availab ...

13 FLIP-FLOP: a stack-oriented multiprocessing system
P. Grabienski
May 1990 **Proceedings of the second annual ACM symposium on Parallel algorithms and architectures**
Publisher: ACM Press
Full text available: pdf(811.11 KB) Additional Information: full citation, references, index terms

14 MIDL - a microinstruction description language
Marleen Sint
December 1981 **ACM SIGMICRO Newsletter , Proceedings of the 14th annual workshop on Microprogramming MICRO 14,** Volume 12 Issue 4
Publisher: IEEE Press, ACM Press
Full text available: pdf(848.63 KB) Additional Information: full citation, abstract, references, citings, index terms

A microinstruction description language called MIDL is introduced. A MIDL description of a microarchitecture defines the semantics and triggering conditions of all microoperations. It also defines operand selection. MIDL incorporates a timing model that allows detailed specification of the timing of each microoperation, and a sequencing model that allows the description of many different sequencing schemes.

15 A proposed high-speed computer design
Trevor Turton
October 1979 **ACM SIGARCH Computer Architecture News,** Volume 7 Issue 10
Publisher: ACM Press
Full text available: pdf(1.16 MB) Additional Information: full citation, abstract, references, citings

The designs of several high performance general purpose computers built during the last two decades are examined. The performance limitations they encountered and their approaches to overcoming these problems are discussed. An alternate design approach is described which avoids these problems. This is to have the computer support the simultaneous execution of several independent programs by multiprogramming its various components at the sub-instruction level. Estimates of its performance are mad ...

16 Efficient vector processing on dataflow supercomputer SIGMA-1
K. Hiraki, S. Sekiguchi, T. Shimada
November 1988 **Proceedings of the 1988 ACM/IEEE conference on Supercomputing**
Publisher: IEEE Computer Society Press
Full text available: pdf(780.87 KB) Additional Information: full citation, abstract, references, citings, index terms

Efficiency in vector handling is the key to obtaining high performance in numerical programs. So far, the main defect of dataflow computers is inefficiency in vector processing. We propose structure-flow processing as a new scheme for handling data structures such as vectors in dataflow architecture. The main objective of structure-flow processing is to enhance vector processing performance of a dataflow computer. In this structure-flow processing scheme, the arrival of a data structure unr ...

17 An intermediate language for source and target independent code optimization

Dennis J. Frailey
August 1979 **ACM SIGPLAN Notices , Proceedings of the 1979 SIGPLAN symposium on Compiler construction SIGPLAN '79**, Volume 14 Issue 8
Publisher: ACM Press
Full text available: pdf(864.12 KB) Additional Information: full citation, abstract, references, index terms

This paper describes an intermediate language to be generated by a syntax analyzer and processed by a code generator. An (essentially) optional code optimization phase may be used before code generation. The language is designed to exclude source and target dependencies (these are specified in a set of auxiliary tables) and has been used to implement a general purpose code optimization module. This module has been incorporated into compilers for several source languages and has resulted in ...

18 OS and compiler considerations in the design of the IA-64 architecture

Rumi Zahir, Jonathan Ross, Dale Morris, Drew Hess
November 2000 **ACM SIGOPS Operating Systems Review , ACM SIGARCH Computer Architecture News , Proceedings of the ninth international conference on Architectural support for programming languages and operating systems ASPLOS-IX**, Volume 34 , 28 Issue 5 , 5
Publisher: ACM Press
Full text available: pdf(90.50 KB) Additional Information: full citation, abstract, references, citings, index terms

Increasing demands for processor performance have outstripped the pace of process and frequency improvements, pushing designers to find ways of increasing the amount of work that can be processed in parallel. Traditional RISC architectures use hardware approaches to obtain more instruction-level parallelism, with the compiler and the operating system (OS) having only indirect visibility into the mechanisms used.The IA-64 architecture [14] was specifically designed to enable systems which create ...

19 OS and compiler considerations in the design of the IA-64 architecture

Rumi Zahir, Jonathan Ross, Dale Morris, Drew Hess
November 2000 **ACM SIGPLAN Notices**, Volume 35 Issue 11
Publisher: ACM Press
Full text available: pdf(1.15 MB) Additional Information: full citation, abstract, references, index terms

Increasing demands for processor performance have outstripped the pace of process and frequency improvements, pushing designers to find ways of increasing the amount of work that can be processed in parallel. Traditional RISC architectures use hardware approaches to obtain more instruction-level parallelism, with the compiler and the operating system (OS) having only indirect visibility into the mechanisms used.The IA-64 architecture [14] was specifically designed to enable systems which create ...

20 MOE: a special-purpose parallel computer for high-speed, large-scale molecular orbital calculation

Koji Hashimoto, Hiroto Tomita, Koji Inoue, Katsuhiko Metsugi, Kazuaki Murakami, Shinjiro Inabata, So Yamada, Nobuaki Miyakawa, Hajime Takashima, Kunihiro Kitamura, Shigeru Obara, Takashi Amisaki, Kazutoshi Tanabe, Umpei Nagashima
January 1999 **Proceedings of the 1999 ACM/IEEE conference on Supercomputing (CDROM)**
Publisher: ACM Press
Full text available: pdf(1.95 MB) Additional Information: full citation, references, index terms

Keywords: ab initio molecular orbital calculations, PPRAM (parallel-processing random-access memory), PPRAM-link, parallel processing, special-purpose computer

Results 1 - 20 of 200 Result page: **1** 2 3 4 5 6 7 8 9 10 next

Terms used
variable lenght operand calculation translation flag

Found 46,426 of 171,143

Sort results by relevance
Display results expanded form
Save results to a Binder
Search Tips
Open results in a new window

Try an Advanced Search
Try this search in The ACM Guide

Results 1 - 20 of 200 Result page: 1 2 3 4 5 6 7 8 9 10 next
Best 200 shown Relevance scale

1 N-synchronous Kahn networks: a relaxed model of synchrony for real-time systems
Albert Cohen, Marc Duranton, Christine Eisenbeis, Claire Pagetti, Florence Plateau, Marc Pouzet
January 2006 **Conference record of the 33rd ACM SIGPLAN-SIGACT symposium on Principles of programming languages POPL'06**
Publisher: ACM Press
Full text available: pdf(225.46 KB) Additional Information: full citation, abstract, references, index terms

The design of high-performance stream-processing systems is a fast growing domain, driven by markets such like high-end TV, gaming, 3D animation and medical imaging. It is also a surprisingly demanding task, with respect to the algorithmic and conceptual simplicity of streaming applications. It needs the close cooperation between numerical analysts, parallel programming experts, real-time control experts and computer architects, and incurs a very high level of quality insurance and optimization. ...

Keywords: correctness by construction, resource constraints, streaming applications, subtyping, synchronous languages

2 Native code compilation of Erlang's bit syntax
Per Gustafsson, Konstantinos Sagonas
October 2002 **Proceedings of the 2002 ACM SIGPLAN workshop on Erlang**
Publisher: ACM Press
Full text available: pdf(196.81 KB) Additional Information: full citation, abstract, references, citings

Erlang's bit syntax caters for flexible pattern matching on bit streams (objects known as *binaries*). Binaries are nowadays heavily used in typical Erlang applications such as protocol programming, which in turn has created a need for efficient support of the basic operations on binaries.To this effect, we describe a scheme for efficient native code compilation of Erlang's bit syntax. The scheme relies on *partial translation* for avoiding code explosion, an ...

3 Efficiently computing static single assignment form and the control dependence graph
Ron Cytron, Jeanne Ferrante, Barry K. Rosen, Mark N. Wegman, F. Kenneth Zadeck
October 1991 **ACM Transactions on Programming Languages and Systems (TOPLAS),** Volume 13 Issue 4

Publisher: ACM Press
Full text available: pdf(2.49 MB) Additional Information: full citation, references, citings, index terms, review

Keywords: control dependence, control flow graph, def-use chain, dominator, optimizing compilers

4 The VAL Language: Description and Analysis
James R. McGraw
January 1982 **ACM Transactions on Programming Languages and Systems (TOPLAS),** Volume 4 Issue 1
Publisher: ACM Press
Full text available: pdf(2.63 MB) Additional Information: full citation, references, citings, index terms

5 Proceedings of the SIGNUM conference on the programming environment for development of numerical software
March 1979 **ACM SIGNUM Newsletter,** Volume 14 Issue 1
Publisher: ACM Press
Full text available: pdf(5.02 MB) Additional Information: full citation

6 Fortran 8X draft
Loren P. Meissner
December 1989 **ACM SIGPLAN Fortran Forum,** Volume 8 Issue 4
Publisher: ACM Press
Full text available: pdf(21.36 MB) Additional Information: full citation, abstract, index terms

Standard Programming Language Fortran. This standard specifies the form and establishes the interpretation of programs expressed in the Fortran language. It consists of the specification of the language Fortran. No subsets are specified in this standard. The previous standard, commonly known as "FORTRAN 77", is entirely contained within this standard, known as "Fortran 8x". Therefore, any standard-conforming FORTRAN 77 program is standard conforming under this standard. New features can b ...

7 Real-time shading
Marc Olano, Kurt Akeley, John C. Hart, Wolfgang Heidrich, Michael McCool, Jason L. Mitchell, Randi Rost
August 2004 **Proceedings of the conference on SIGGRAPH 2004 course notes GRAPH '04**
Publisher: ACM Press
Full text available: pdf(7.39 MB) Additional Information: full citation, abstract

Real-time procedural shading was once seen as a distant dream. When the first version of this course was offered four years ago, real-time shading was possible, but only with one-of-a-kind hardware or by combining the effects of tens to hundreds of rendering passes. Today, almost every new computer comes with graphics hardware capable of interactively executing shaders of thousands to tens of thousands of instructions. This course has been redesigned to address today's real-time shading capabili ...

8 Algorithm 719; Multiprecision translation and execution of FORTRAN programs
David H. Bailey

September 1993 **ACM Transactions on Mathematical Software (TOMS)**, Volume 19 Issue 3
Publisher: ACM Press
Full text available: pdf(2.03 MB) Additional Information: full citation, abstract, references, citings, index terms

This paper describes two Fortran utilities for multiprecision computation. The first is a package of Fortran subroutines that perform a variety of arithmetic operations and transcendental functions on floating point numbers of arbitrarily high precision. This package is in some cases over 200 times faster than that of certain other packages that have been developed for this purpose. The second utility is a translator program, which facilitates the conversion of ordinary Fortran p ...

Keywords: multiple-precision computation, multiprecision arithmetic

9 An optimizing compiler for lexically scoped LISP
Rodney A. Brooks, Richard P. Gabriel, Guy L. Steele
June 1982 **ACM SIGPLAN Notices , Proceedings of the 1982 SIGPLAN symposium on Compiler construction SIGPLAN '82**, Volume 17 Issue 6
Publisher: ACM Press
Full text available: pdf(1.37 MB) Additional Information: full citation, abstract, references, citings, index terms

We are developing an optimizing compiler for a dialect of the LISP language. The current target architecture is the S-I, a multiprocessing supercomputer designed at Lawrence Livermore National Laboratory. While LISP is usually thought of as a language primarily for symbolic processing and list manipulation, this compiler is also intended to compete with the S-1 PASCAL and FORTRAN compilers for quality of compiled numerical code. The S-1 is designed for extremely high-speed signal processing ...

10 Migrating a CISC computer family onto RISC via object code translation
Kristy Andrews, Duane Sand
September 1992 **ACM SIGPLAN Notices , Proceedings of the fifth international conference on Architectural support for programming languages and operating systems ASPLOS-V**, Volume 27 Issue 9
Publisher: ACM Press
Full text available: pdf(1.13 MB) Additional Information: full citation, references, citings, index terms

11 A practical method for code generation based on exhaustive search
David W. Krumme, David H. Ackley
June 1982 **ACM SIGPLAN Notices , Proceedings of the 1982 SIGPLAN symposium on Compiler construction SIGPLAN '82**, Volume 17 Issue 6
Publisher: ACM Press
Full text available: pdf(1.10 MB) Additional Information: full citation, abstract, references, citings, index terms

An original method for code generation has been developed in conjunction with the construction of a compiler for the C programming language on the DEC-10 computer. The method is comprehensive, determining evaluation order and doing register allocation and instruction selection simultaneously. It uses exhaustive search rather than heuristics, and is table-driven, with most machine-specific information isolated in the tables. Testing and evaluation have shown that the method is effective, tha ...

12 The design of a virtual machine for Ada
L. J. Groves, W. J. Rogers

November 1980 **ACM SIGPLAN Notices , Proceeding of the ACM-SIGPLAN symposium on Ada programming language SIGPLAN '80**, Volume 15 Issue 11
Publisher: ACM Press
Full text available: pdf(1.27 MB) Additional Information: full citation, abstract, references

An implementation of Ada should be based on a machine-independent translator generating code for a Virtual Machine, which can be realised on a variety of machines. This approach, which leads to a high degree of compiler portability, has been very successful in a number of recent language implementation projects and is the approach which has been specified by the U. S. Army and Air Force in their requirements for Ada implementations.This paper discusses the rationale, requirements and design of s ...

13 A hardware architecture for implementing protection rings
Michael D. Schroeder, Jerome H. Saltzer
March 1972 **Communications of the ACM**, Volume 15 Issue 3
Publisher: ACM Press
Full text available: pdf(1.50 MB) Additional Information: full citation, abstract, references, citings

Protection of computations and information is an important aspect of a computer utility. In a system which uses segmentation as a memory addressing scheme, protection can be achieved in part by associating concentric rings of decreasing access privilege with a computation. This paper describes hardware processor mechanisms for implementing these rings of protection. The mechanisms allow cross-ring calls and subsequent returns to occur without trapping to the supervisor. Automatic hardware v ...

Keywords: Multics, access control, computer utility, hardware access control, protection, protection hardware, protection rings, segmentation, shared information, time-sharing, virtual memory

14 Parsing and compiling using Prolog
Jacques Cohen, Timothy J. Hickey
March 1987 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 9 Issue 2
Publisher: ACM Press
Full text available: pdf(2.83 MB) Additional Information: full citation, abstract, references, citings, index terms, review

This paper presents the material needed for exposing the reader to the advantages of using Prolog as a language for describing succinctly most of the algorithms needed in prototyping and implementing compilers or producing tools that facilitate this task. The available published material on the subject describes one particular approach in implementing compilers using Prolog. It consists of coupling actions to recursive descent parsers to produce syntax-trees which are subsequently utilized ...

15 Cache Memories
Alan Jay Smith
September 1982 **ACM Computing Surveys (CSUR)**, Volume 14 Issue 3
Publisher: ACM Press
Full text available: pdf(4.61 MB) Additional Information: full citation, references, citings, index terms

16 A 32-bit CMOS microprocessor with six-stage pipeline structure
H. Kaneko, Y. Miki, S. Nohara, K. Koya, M. Araki
November 1986 **Proceedings of 1986 ACM Fall joint computer conference**
Publisher: IEEE Computer Society Press

Full text available: pdf(831.34 KB) Additional Information: full citation, references, citings, index terms

17 GPGPU: general purpose computation on graphics hardware

David Luebke, Mark Harris, Jens Krüger, Tim Purcell, Naga Govindaraju, Ian Buck, Cliff Woolley, Aaron Lefohn

August 2004 **Proceedings of the conference on SIGGRAPH 2004 course notes GRAPH '04**

Publisher: ACM Press

Full text available: pdf(63.03 MB) Additional Information: full citation, abstract

The graphics processor (GPU) on today's commodity video cards has evolved into an extremely powerful and flexible processor. The latest graphics architectures provide tremendous memory bandwidth and computational horsepower, with fully programmable vertex and pixel processing units that support vector operations up to full IEEE floating point precision. High level languages have emerged for graphics hardware, making this computational power accessible. Architecturally, GPUs are highly parallel s ...

18 An Elementary Discussion of Compiler/Interpreter Writing

R. L. Glass

March 1969 **ACM Computing Surveys (CSUR)**, Volume 1 Issue 1

Publisher: ACM Press

Full text available: pdf(1.85 MB) Additional Information: full citation, references, citings, index terms

19 Translator writing systems

Jerome Feldman, David Gries

February 1968 **Communications of the ACM**, Volume 11 Issue 2

Publisher: ACM Press

Full text available: pdf(4.47 MB) Additional Information: full citation, abstract, references, citings

A critical review of recent efforts to automate the writing of translators of programming languages is presented. The formal study of syntax and its application to translator writing are discussed in Section II. Various approaches to automating the postsyntactic (semantic) aspects of translator writing are discussed in Section III, and several related topics in Section IV.

Keywords: compiler compiler-compiler, generator, macroprocessor, meta-assembler, metacompiler, parser, semantics, syntactic analysis, syntax, syntax-directed, translator, translator writing system

20 High speed compilation of efficient object code

C. W. Gear

August 1965 **Communications of the ACM**, Volume 8 Issue 8

Publisher: ACM Press

Full text available: pdf(882.61 KB) Additional Information: full citation, abstract, references, citings, index terms

A three-pass compiler with the following properties is briefly described: The last two passes scan an intermediate language produced by the preceding pass in essentially the reverse of the order in which it was generated, so that the first pass is the only one which has to read the relatively bulky problem-oriented input. The double scan, one in either direction, performed by the first two passes, allows the compiler to remove locally constant expressions and recursively calculable expressi ...

Results 1 - 20 of 200 Result page: **1** 2 3 4 5 6 7 8 9 10 next

PALM INTRANET

Day : Wednesday
Date: 3/8/2006
Time: 18:28:22

Inventor Name Search Result

Your Search was:

Last Name = HILTON
First Name = RONALD

Application#	Patent#	Status	Date Filed	Title	Inventor Name
09992120	Not Issued	71	11/14/2001	State-specific variants of translated code under emulation	HILTON, RONALD
09992121	Not Issued	71	11/14/2001	Flexible caching of translated code under emulation	HILTON, RONALD
09992130	Not Issued	41	11/14/2001	Processing of self-modifying code under emulation	HILTON, RONALD
09992137	Not Issued	71	11/14/2001	Memory address prediction under emulation	HILTON, RONALD
08798305	5889639	150	02/07/1997	PLAIN CARBON STEEL SHUTTER FOR REMOVABLE DATA STORAGE CARTRIDGES	HILTON, RONALD A.
09201248	Not Issued	161	11/30/1998	DIGITAL PHONE SYSTEM	HILTON, RONALD D.
09201460	Not Issued	161	11/30/1998	METHOD AND APPARATUS FOR DYNAMIC DOMAIN NAMES	HILTON, RONALD D.
60067231	Not Issued	159	12/02/1997	METHOD AND APPARATUS FOR DYNAMIC DOMAIN NAMES	HILTON, RONALD D.
60067233	Not Issued	159	12/02/1997	DIGITAL PHONE SYSTEM	HILTON, RONALD D.
08796271	5896241	150	02/07/1997	PLAIN CARBON STEEL HUB FOR DATA STORAGE DEVICE	HILTON, RONALD L.
09211954	6292996	150	12/15/1998	METHOD OF MAKING A PLAIN CARBON STEEL HUB FOR DATA STORAGE DEVICE	HILTON, RONALD L.
11254290	Not Issued	30	10/19/2005	Processing of self-modifying code in multi-address-space and multi-processor systems	HILTON, RONALD N.
11254291	Not Issued	30	10/19/2005	Queue or stack based cache entry reclaim method	HILTON, RONALD N.
11271075	Not Issued	20	11/10/2005	Sparse table compaction method	HILTON, RONALD N.
11271681	Not Issued	20	11/10/2005	Peer-based partitioning method for system resource sharing	HILTON, RONALD N.
11280554	Not Issued	20	11/15/2005	Distributed shared I/O cache subsystem	HILTON, RONALD N.
60620364	Not Issued	159	10/19/2004	Processing of self-modifying code in multi-address-space and multi-processor systems	HILTON, RONALD N.
60620365	Not Issued	159	10/19/2004	Queue or stack based cache entry reclaim method	HILTON, RONALD N.
60628332	Not Issued	159	11/15/2004	Distributed shared I/O cache subsystem	HILTON, RONALD N.
60628420	Not Issued	159	11/15/2004	Peer-based partitioning method for system resource	HILTON, RONALD N.
60628452	Not Issued	159	11/15/2004	Sparse table compaction method	HILTON, RONALD N.
07816959	Not	166	01/03/1992	S-UNIT ERROR HISTORY INHIBIT (EHI)	HILTON, RONALD N.

	Issued			FACILITY	
07949583	5410668	150	09/23/1992	RECONFIGURABLE CACHE MEMORY WHICH CAN SELECTIVELY INHIBIT ACCESS TO DAMAGED SEGMENTS IN THE CACHE MEMORY	HILTON, RONALD N.
07950459	Not Issued	161	09/24/1992	CONCURRENT BRANCH PROCESSING WITH DUAL INSTRUCTION DECODE	HILTON, RONALD N.
07954297	Not Issued	166	09/30/1992	COMPUTER SYSTEM HAVING CACHE MEMORIES WITH INDEPENDENTLY VALIDATED KEYS IN THE TLB	HILTON, RONALD N.
07993082	5488706	150	12/18/1992	A RETRY REQUEST SYSTEM IN A PIPELINE DATA PROCESSING SYSTEM WHERE EACH REQUESTING UNIT PRESERVERS THE ORDER OF REQUESTS	HILTON, RONALD N.
08033415	Not Issued	161	03/18/1993	S-UNIT ERROR HISTORY INHIBIT (EHI) FACILITY	HILTON, RONALD N.
08337133	5603008	150	11/10/1994	COMPUTER SYSTEM HAVING CACHE MEMORIES WITH INDEPENDENTLY VALIDATED KEYS IN THE TLB	HILTON, RONALD N.

Inventor Search Completed: No Records to Display.

Search Another: Inventor — Last Name: HILTON — First Name: RONALD — Search

To go back use Back button on your browser toolbar.

Back to PALM | ASSIGNMENT | OASIS | Home page